R01-26



A01-29253

AIAA-2001-2067

# COMPUTATIONAL AERODYNAMICS OF A PARATROOPER SEPARATING FROM AN AIRCRAFT

Victor Udoewa*, Ryan Keedy§, Tayfun Tezduyar†
*Team for Advanced Flow Simulation and Modeling (T*AFSM)‡*
*Mechanical Engineering and Materials Science*
*Rice University - MS 321, 6100 Main Street*
*Houston, TX 77005, USA*

Tomoyasu Nonoshita
*Sophia University, Tokyo, JAPAN*

Keith Stein¶, Richard Benney**
*U.S. Army Soldier and Biological Chemical Command*
*Soldier Systems Center - Natick, MA 01760, USA*

and

Andrew Johnson††
*Network Computing Services, 1200 Washington Avenue*
*Minneapolis, MN 55415, USA*

## ABSTRACT

We present the computational methods we have developed for simulation of the aerodynamics of a paratrooper during the time period following immediately after the paratrooper jumps from a cargo aircraft. These methods can also be used for the aerodynamic simulation of a payload, such as a crate of emergency aid or a ground vehicle, being dropped from the rear door of a cargo aircraft. These are applications with major significance in the area of airdrop technology. In both of these cases, the computational challenge is to predict the dynamic behavior and path of the object separating from the aircraft, so that this early stage of the deployment process is successful. The methods we developed to address this challenge are based on the Deforming-Spatial-Domain/Stabilized Space-Time formulation, advanced mesh update methods, and parallel computing on distributed memory parallel supercomputers. The preliminary results for the paratrooper deployment and cargo drop demonstrate that these methods can potentially play a major role in simulation of airdrop systems.

*Graduate Student

§Undergraduate Student

†James F. Barbour Professor, Corresponding Author

‡http://www.mems.rice.edu/TAFSM/

¶Aerospace Engineer

**Aerospace Engineer, Senior Member

††Scientific Applications Development Engineer



## INTRODUCTION

When a cargo aircraft with relatively new design is put into operation, the aerodynamic interaction between the aircraft and a paratrooper separating from that aircraft becomes a very relevant issue in terms of successful and effective deployment of paratroopers. In this paper, we consider a single paratrooper exiting the aircraft. The purpose is to calculate the path of the paratrooper relative to the aircraft, after his exit from the aircraft but prior to the opening of the parachute, while being subjected to aerodynamical forces in addition to gravity. This simulation was first reported in.[1] An improved geometric model for the aircraft, with wing flaps and winglets, was reported in.[2] The computation with this improved model of the aircraft and a more realistic model of the paratrooper was reported in.[3] The simulation is based on solving the Navier-Stokes equations of incompressible flows for flow around both the aircraft and the paratrooper while their relative positions are changing, coupled with solving the equations of motion for the paratrooper. A related problem with very similar computational challenges is separation of a large payload, such as a crate of emergency aid or a ground vehicle, being dropped from the rear door of a cargo aircraft. The Deforming-Spatial-Domain/Stabilized Space-Time (DSD/SST) formulation[4-6] is used for solving this class of problems where the spatial domain occupied by air is changing in time.

The DSD/SST formulation was developed as a general-purpose tool for computation of flows with moving boundaries and interfaces, such as two-fluid and free-surface flows and fluid-structure and fluid-object interactions. This method, inherently, takes into account the motion of the boundaries and interfaces. At each time step of a computation, the locations of the boundaries and interfaces are calculated as part of the overall solution.

Stabilized space-time finite element formulations were used earlier by other researchers in the context of problems with fixed spatial domains (see for example[7]). The DSD/SST formulation is based on the Galerkin/Least-Squares (GLS) formulation.[8] The Streamline-Upwind/Petrov-Galerkin (SUPG)[9,10] ans Pressure-Stabilizing/Petrov-Galerkin (PSPG)[4] formulations are the essential components of the GLS formulation. When these stabilized formulations are implemented with a sound understanding of their underlying concepts, flows at high Reynolds numbers can be computed without introducing excessive numerical dissipation, and equal-order interpolations functions for velocity and pressure can be used to simplify the implementation.

Another component of the DSD/SST formulation is an automatic mesh moving technique, which is used for updating the mesh as the spatial domain occupied by the fluid changes its shape every time step. This mesh moving method, first introduced in,[11] is based on moving the nodal points as governed by the equations of linear elasticity. The motion of the internal nodes is determined by solving these additional equations, with the boundary conditions for these mesh motion equations specified in such a way that they match the normal velocity of the fluid at the solid surface. Reducing the frequency of remeshing is the most important consideration taken into account in the development of the mesh moving method. We have developed and implemented a number of ideas to that end, such as giving smaller elements more protection against excessive element distortion, as originally reported in.[11]

Discretization of the class of problems addressed here results in large, coupled nonlinear equation systems that need to be solved at every time step. The coupling between the blocks of equations corresponding to the Navier-Stokes and mesh moving equations are handled in an iterative fashion. At each nonlinear iteration, the vectors of unknowns associated with these two blocks of equations are updated individually, based on the Newton-Raphson method. While updating the vector of unknowns associated with one of the blocks, we use the most recently updated values of the vectors of unknowns associated with the other block. The coupled, linear equation system that needs to be solved in updating each of the two vectors of unknowns is also solved iteratively, with the GMRES search technique.[12] We have implemented these solution techniques for distributed-memory parallel computing, and the results reported here were obtained by carrying out the computations on a CRAY T3E-1200.

In this paper, we first review the governing equations and the DSD/SST formulation. Then we describe the mesh update techniques used in conjunction with the DSD/SST formulation, and present results from preliminary computations. Examples presented are paratrooper deployment and cargo drop from a transport aircraft.

## GOVERNING EQUATIONS

Let $\Omega_t \subset R^{n_{sd}}$ be the spatial fluid mechanics domain with boundary $\Gamma_t$ at time $t \in (0,T)$, where the subscript $t$ indicates the time-dependence of the spatial domain and its boundary. The Navier-Stokes equations of incompressible flows can be written on $\Omega_t$ and $\forall t \in (0,T)$ as

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} - \mathbf{f}\right) - \nabla\cdot\boldsymbol{\sigma} = 0, \qquad (1)$$

$$\nabla\cdot\mathbf{u} = 0, \qquad (2)$$

where $\rho$, $\mathbf{u}$ and $\mathbf{f}$ are the density, velocity and the external force, respectively. The stress tensor $\boldsymbol{\sigma}$ is defined as

$$\boldsymbol{\sigma}(p,\mathbf{u}) = -p\mathbf{I} + 2\mu\boldsymbol{\varepsilon}(\mathbf{u}). \qquad (3)$$

Here $p$, $\mathbf{I}$ and $\mu$ are the pressure, identity tensor and the viscosity, respectively. The strain rate tensor $\boldsymbol{\varepsilon}(\mathbf{u})$ is defined as

$$\boldsymbol{\varepsilon}(\mathbf{u}) = \frac{1}{2}\left((\nabla\mathbf{u}) + (\nabla\mathbf{u})^T\right). \qquad (4)$$

Both Dirichlet- and Neumann-type boundary conditions are accounted for:

$$\begin{aligned} \mathbf{u} &= \mathbf{g} \text{ on } (\Gamma_t)_g, \\ \mathbf{n}\cdot\boldsymbol{\sigma} &= \mathbf{h} \text{ on } (\Gamma_t)_h. \end{aligned} \qquad (5)$$

Here $(\Gamma_t)_g$ and $(\Gamma_t)_h$ are complementary subsets of the boundary $\Gamma_t$, $\mathbf{n}$ is the unit normal vector at the boundary, and $\mathbf{g}$ and $\mathbf{h}$ are given functions. A divergence-free velocity field is specified as the initial condition.

## DEFORMING-SPATIAL-DOMAIN/ STABILIZED SPACE-TIME (DSD/SST) FORMULATION

In discretization of the space-time domain, the time interval $(0,T)$ is partitioned into subintervals $I_n = (t_n, t_{n+1})$, where $t_n$ and $t_{n+1}$ belong to an ordered series of time levels $0 = t_0 < t_1 \cdots < t_N = T$. Let $\Omega_n = \Omega_{t_n}$ and $\Gamma_n = \Gamma_{t_n}$ to simplify the notation. The space-time slab $Q_n$ is defined as the domain enclosed by the surfaces $\Omega_n$, $\Omega_{n+1}$, and $P_n$, where $P_n$ is the lateral surface of $Q_n$ described by the boundary $\Gamma_n$ as $t$ traverses $I_n$ (see Figure 1).

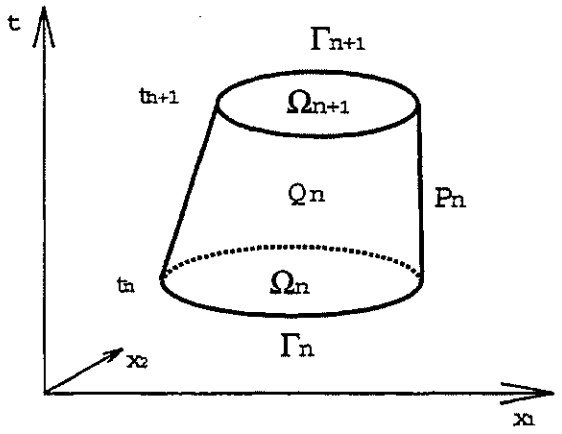


Figure 1. Space-time concept.

The Dirichlet- and Neumann-type boundary conditions are specified over $(P_n)_g$ and $(P_n)_h$. For this discretization, the finite element trial function spaces $(\mathcal{S}_{\mathbf{u}}^h)_n$ for velocity and $(\mathcal{S}_p^h)_n$ for pressure, and the corresponding test function spaces $(\mathcal{V}_{\mathbf{u}}^h)_n$ and $(\mathcal{V}_p^h)_n$ are defined as follows:

$$(\mathcal{S}_{\mathbf{u}}^h)_n = \{\mathbf{u}^h | \mathbf{u}^h \in [H^{1h}(Q_n)]^{n_{sd}}, \mathbf{u}^h \doteq \mathbf{g}^h \text{ on } (P_n)_g\}, \quad (6)$$

$$(\mathcal{V}_{\mathbf{u}}^h)_n = \{\mathbf{w}^h | \mathbf{w}^h \in [H^{1h}(Q_n)]^{n_{sd}}, \mathbf{w}^h \doteq \mathbf{0} \text{ on } (P_n)_g\}, \quad (7)$$

$$(\mathcal{S}_p^h)_n = (\mathcal{V}_p^h)_n = \{q^h | q^h \in H^{1h}(Q_n)\}. \quad (8)$$

Here $H^{1h}(Q_n)$ is the finite-dimensional function space over the space-time slab $Q_n$. Over the element domain, this space is formed by using first-order polynomials in both space and time. The interpolation functions are continuous in space but discontinuous in time.

The DSD/SST formulation is written as follows: given $(\mathbf{u}^h)_n^-$, find $\mathbf{u}^h \in (\mathcal{S}_{\mathbf{u}}^h)_n$ and $p^h \in (\mathcal{S}_p^h)_n$ such that $\forall \mathbf{w}^h \in (\mathcal{V}_{\mathbf{u}}^h)_n$ and $q^h \in (\mathcal{V}_p^h)_n$:

$$\begin{aligned}
&\int_{Q_n} \mathbf{w}^h \cdot \rho \left( \frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \nabla \mathbf{u}^h - \mathbf{f}^h \right) dQ \\
&+ \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}(p^h, \mathbf{u}^h) dQ \\
&- \int_{(P_n)_h} \mathbf{w}^h \cdot \mathbf{h}^h dP \\
&+ \int_{Q_n} q^h \nabla \cdot \mathbf{u}^h dQ \\
&+ \int_{\Omega_n} (\mathbf{w}^h)_n^+ \cdot \rho \left( (\mathbf{u}^h)_n^+ - (\mathbf{u}^h)_n^- \right) d\Omega \\
&+ \sum_{e=1}^{(n_{el})_n} \int_{Q_n^e} \frac{\tau_{\text{LSME}}}{\rho} \mathbf{L}(q^h, \mathbf{w}^h) \cdot \\
&\qquad \left[ \mathbf{L}(p^h, \mathbf{u}^h) - \rho \mathbf{f}^h \right] dQ \\
&+ \sum_{e=1}^{n_{el}} \int_{Q_n^e} \tau_{\text{LSIC}} \nabla \cdot \mathbf{w}^h \rho \nabla \cdot \mathbf{u}^h dQ \\
&= 0, \qquad (9)
\end{aligned}$$

where

$$\mathbf{L}(q^h, \mathbf{w}^h) = \rho \left( \frac{\partial \mathbf{w}^h}{\partial t} + \mathbf{u}^h \cdot \nabla \mathbf{w}^h \right) - \nabla \cdot \boldsymbol{\sigma}(q^h, \mathbf{w}^h). \quad (10)$$

This formulation is sequentially applied to all space-time slabs $Q_0, Q_1, Q_2, \ldots, Q_{N-1}$. The computation starts with

$$(\mathbf{u}^h)_0^- = \mathbf{u}_0, \quad \nabla \cdot \mathbf{u}_0 = 0 \quad \text{on } \Omega_0. \quad (11)$$

Here $\tau_{\text{LSME}}$ and $\tau_{\text{LSIC}}$ are the stabilization parameters (see[2,13]). For an earlier, detailed reference on this stabilized formulation see.[4]

## MESH UPDATE METHOD

In application of the the DSD/SST method to flows with moving objects, as the computations proceed, the mesh needs to be updated to accommodate the changes in the spatial domain occupied by the fluid. This needs to be accomplished as efficiently as possible, and without compromising the accuracy of the solution. Selection of the way to update the mesh would depend on the geometric complexity of the moving objects, the complexity of the overall problem geometry, and how the initial mesh was generated. In general, the mesh update could have two

components: moving the mesh for as long as possible and remeshing (i.e. generating fully or partially a new set of nodes and elements) whenever the element distortion becomes too high.

Most real-world problems involve complex geometries. A complex geometry typically requires an automatic mesh generator. Earlier we developed our own automatic mesh generator so that we could have a number of special features, such as structured layers of elements around solid surfaces and high-speed mesh generation. This automatic, 3D mesh generator is described in.[14] It has been used very effectively in large a number of simulations (for early examples see[1,15]). With this mesh generator, with its capability to build structured layers of elements around solid objects with reasonable geometric complexity, we can fully control the mesh resolution near solid objects. This feature can be used for more accurate representation of the boundary layers. The mesh generator also has the capability to generate meshes for fluid-object interactions in spatially periodic flows (see[16]).

Automatic mesh generation might become a prohibitively costly task when the number of elements becomes very large or when the frequency of remeshing has to be high. As an alternative, sometimes, special-purpose mesh generators could be designed specifically for a certain class of problems. Depending on the complexity of the problem, this alternative might involve a high initial design cost, but then the mesh generation cost becomes minimal. In fact we selected this path in a number of our earlier simulations and were able to overcome the mesh generation issues very effectively (see for example[17]).

In mesh moving, on a solid surface the normal velocity of the mesh has to match the normal velocity of the fluid. Provided that this condition is satisfied, the mesh can be moved in any way desired, with the main objective being to reduce the frequency of remeshing. The cost of automatic 3D mesh generation is a major reason for trying to reduce the frequency of remeshing. Furthermore, after every time we remesh, we need to project the solution from the old mesh to the new one. Not only does this introduce projection errors, but the computing time consumed by this projection is rather significant in 3D simulations. All these factors make a strong case for designing mesh update techniques that reduce the frequency of remeshing.

In some cases, when the changes in the shape of the fluid domain allow it, a special-purpose mesh moving method can be used in conjunction with a special-purpose mesh generator. Then the simulations can be carried out without solving any additional equations to determine the motion of the mesh and without remeshing. One of our earliest simulations in this category, 3D computation of sloshing in a vertically vibrating container, can be found in.[18]

For general-purpose mesh moving, we developed an automatic mesh moving technique,[11] where the nodal motions are governed by the equations of elasticity. The motion of the internal nodes is determined by solving these additional equations, with the boundary conditions for these equations specified in such a way that they match the normal velocity of the solid surface.

The special mesh update method developed for the paratrooper deployment and payload drop is based on automatic mesh moving and remeshing as needed. However, the mesh moving and remeshing is limited to a mesh zone around the paratrooper or payload and covering their expected trajectory. The equations of elasticity are solved only for the nodes in this zone, and when the mesh distortion in this zone becomes too high, remeshing takes place only in this zone. This approach significantly reduces the computational cost associated with mesh moving and remeshing.

## SIMULATIONS

The aircraft is traveling at 130 knots, with an angle of attack of 10 degrees. For simulations of a paratrooper jumping from a side door, we assume symmetry with respect to the vertical plane passing through the middle of the aircraft. For the payload drop, we model the full aircraft. The original aircraft model was introduced in.[1] Here we use an improved version of that model by including the wing flaps and winglets. In defining the parameters of the problem with non-dimensional numbers, the length of the aircraft (nose to tail wingtip) is taken as 8.81 units, and the free-stream velocity as 1.00 unit. The boundary conditions are uniform upstream velocity, no-slip on the aircraft surface, zero normal velocity and zero shear stress at the crossflow and side boundaries, and a traction-free condition at the outflow boundary. For the engines, the intake and exhaust flow velocity profiles are assumed to be uniform, and prescribed as 1.0 and 3.0, respectively. No-slip conditions are used on the surface of the paratrooper and payload. The tetrahedral mesh used in the paratrooper deployment simulation has 129,090 nodes and 728,902 elements, while the mesh used in the cargo drop simulation has 332,498 nodes and 1,952,559 elements.

We realize that these mesh refinement levels are below the levels required to secure quantitatively dependable solutions for the complex flow problems we are addressing here. However, we see these preliminary computations as demonstrations of the tools we have been developing for this class of problems, and also as a way of having a general, qualitative understanding of the solution in each problem. The parallel computations were carried on a CRAY T3E-1200. Figure 2 shows the aircraft and the paratrooper in the early stages of the separation. Figure 3 shows the close up view of the flow field around the paratrooper. Figure 4 shows the payload as it exits through the cargo door.

## CONCLUDING REMARKS

We described the computational methods we have developed for simulation of the aerodynamics of a paratrooper during the time period following immediately after the paratrooper jumps from a cargo aircraft. Although the paratrooper deployment was our main purpose in developing these methods, they can also be used for simulation of related problems with very similar computational challenges such as separation of a large payload being dropped from the rear door of a cargo aircraft. The payload can, for example, be a crate of emergency aid or a ground vehicle. The computational methods are based on the Deforming-Spatial-Domain/Stabilized Space-Time formulation, advanced mesh update methods, and parallel computing on distributed memory parallel supercomputers. With the results we obtained from our preliminary computations for the paratrooper deployment and cargo drop, we demonstrated that these methods can potentially play a major role in simulation of airdrop systems. Also as a result of this work, we were able to gain some understanding of the main features of the dynamics and fluid mechanics involved in the problems addressed.

## ACKNOWLEDGEMENT

This work was supported by NASA JSC (grant no. NAG9-1059) and by the US Army Natick Soldier Center (contract no. DAAD16-00-C-9222). The content does not necessarily reflect the position or the policy of the government, and no official endorsement should be inferred.

## References

[1] T.E. Tezduyar, S. Aliabadi, M. Behr, A. Johnson, V. Kalro, and M. Litke, "Flow simulation and high performance computing", *Computational Mechanics*, **18** (1996) 397–412.

[2] T.E. Tezduyar and Y. Osawa, "Methods for parallel computation of complex flow problems", *Parallel Computing*, **25** (1999) 2039–2066.

[3] V. Udoewa, R. Keedy, T. Nonoshita, T. Tezduyar, K. Stein, and A. Johnson, "Aerodynamic simulation of an object separating from an aircraft during initial deployment", to appear in *Proceedings of the First MIT Conference on Computational Fluid and Solid Mechanics*, Boston, Massachusetts, 2001.

[4] T.E. Tezduyar, "Stabilized finite element formulations for incompressible flow computations", *Advances in Applied Mechanics*, **28** (1991) 1–44.

[5] T.E. Tezduyar, M. Behr, and J. Liou, "A new strategy for finite element computations involving moving boundaries and interfaces – the deforming-spatial-domain/space-time procedure: I. The concept and the preliminary tests", *Computer Methods in Applied Mechanics and Engineering*, **94** (1992) 339–351.

[6] T.E. Tezduyar, M. Behr, S. Mittal, and J. Liou, "A new strategy for finite element computations involving moving boundaries and interfaces – the deforming-spatial-domain/space-time procedure: II. Computation of free-surface flows, two-liquid flows, and flows with drifting cylinders", *Computer Methods in Applied Mechanics and Engineering*, **94** (1992) 353–371.

[7] T.J.R. Hughes and G.M. Hulbert, "Space-time finite element methods for elastodynamics: formulations and error estimates", *Computer Methods in Applied Mechanics and Engineering*, **66** (1988) 339–363.

[8] T.J.R. Hughes, L.P. Franca, and G.M. Hulbert, "A new finite element formulation for computational fluid dynamics: VIII. the Galerkin/least-squares method for advective-diffusive equations", *Computer Methods in Applied Mechanics and Engineering*, **73** (1989) 173–189.

[9] T.J.R. Hughes and A.N. Brooks, "A multidimensional upwind scheme with no crosswind diffusion", in T.J.R. Hughes, editor, *Finite Element Methods for Convection Dominated Flows*, AMD-Vol.34, 19–35, ASME, New York, 1979.

[10] A.N. Brooks and T.J.R. Hughes, "Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations", *Computer Methods in Applied Mechanics and Engineering*, **32** (1982) 199–259.

[11] T.E. Tezduyar, M. Behr, S. Mittal, and A.A. Johnson, "Computation of unsteady incompressible flows with the finite element methods – space-time formulations, iterative strategies and massively parallel implementations", in P. Smolinski, W.K. Liu, G. Hulbert, and K. Tamma, editors, *New Methods in Transient Analysis*, AMD-Vol.143, ASME, New York, (1992) 7–24.

[12] Y. Saad and M. Schultz, "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems", *SIAM Journal of Scientific and Statistical Computing*, **7** (1986) 856–869.

[13] M. Behr and T.E. Tezduyar, "Finite element solution strategies for large-scale flow simulations", *Computer Methods in Applied Mechanics and Engineering*, **112** (1994) 3–24.

[14] A.A. Johnson and T.E. Tezduyar, "Parallel computation of incompressible flows with complex geometries", *International Journal for Numerical Methods in Fluids*, **24** (1997) 1321–1340.

[15] A.A. Johnson and T.E. Tezduyar, "Simulation of multiple spheres falling in a liquid-filled tube", *Computer Methods in Applied Mechanics and Engineering*, **134** (1996) 351–373.

[16] A.A. Johnson and T.E. Tezduyar, "Methods for 3D computation of fluid-object interactions in spatially-periodic flows", to appear in *Computer Methods in Applied Mechanics and Engineering*, 2000.

[17] S. Mittal and T.E. Tezduyar, "Massively parallel finite element computation of incompressible flows involving fluid-body interactions", *Computer Methods in Applied Mechanics and Engineering*, **112** (1994) 253–282.

[18] T. Tezduyar, S. Aliabadi, M. Behr, A. Johnson, and S. Mittal, "Parallel finite-element computation of 3D flows", *IEEE Computer*, **26** (1993) 27–36.

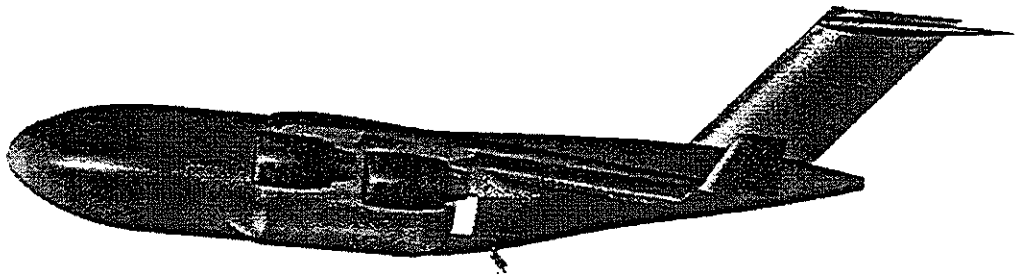

Figure 2. Paratrooper separation from the aircraft. The colors on the paratrooper and aircraft surface show the air pressure distribution.

Figure 3. Flow around the paratrooper. The streamlines and the air pressure distribution on the paratrooper.

Figure 4. Payload exiting through the cargo door. The stream ribbons and the air pressure distribution on the aircraft and payload surfaces.